University of West Florida

Securing Control Systems with MultiLayer Static Mutation

Sharon Simmons, Dennis Edwards, Norman Wilde

FLORIDA STATE UNIVERSITY · 1851 · VIRES ARTES MORES

CENTER FOR ADVANCED POWER SYSTEMS

UNF

Why More Security?

Increased automation

- More agent-based control systems
- Control systems vulnerable

SCADA focuses on prevention

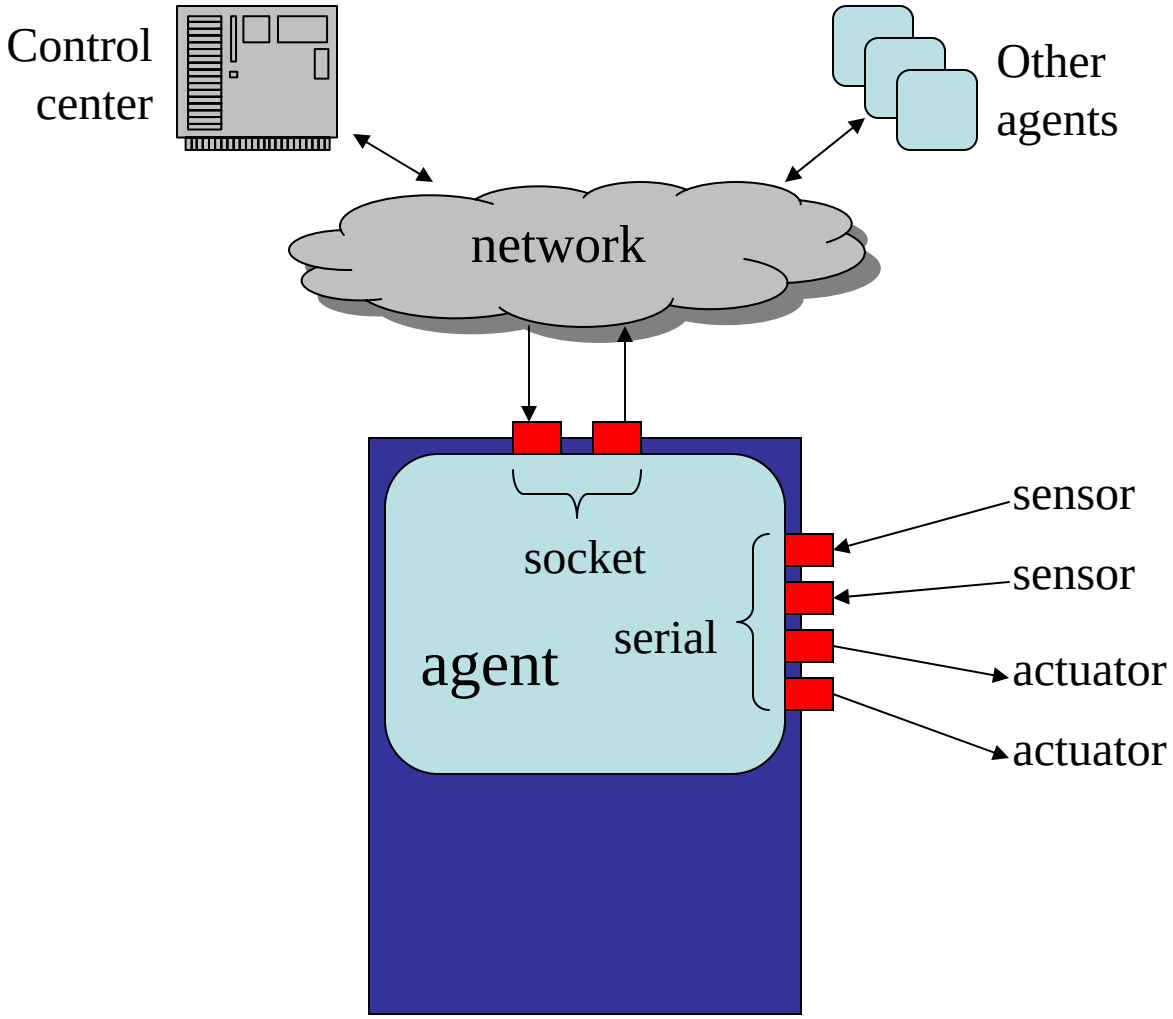
- Concentrates on authentication, validation
- Eventually, hacker will get through

MultiLayer Security Model (MLSM)

- Focuses on detection and recovery
- Used with SCADA, encryption, etc.
- Prototype recovers from prevalent attack method
- Completes the prevention, detection, recovery triad of security

University of West Florida

Assumed Control Element Structure



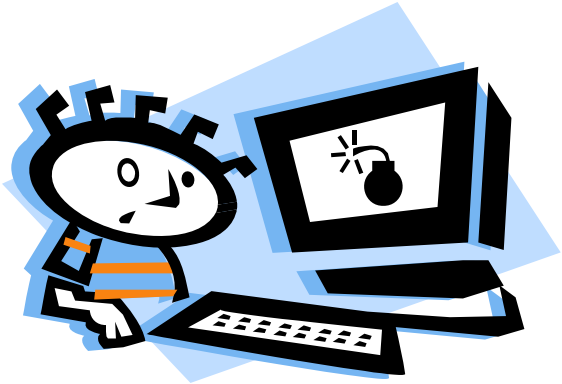
Consequences & Failure Modes

- **Successful attack leads to consequence**
- **Consequences can be categorized as failure mode**
- **Three failure modes**
  - Crash failure
  - Response failure
  - Byzantine failure

University of West Florida

Crash Failure

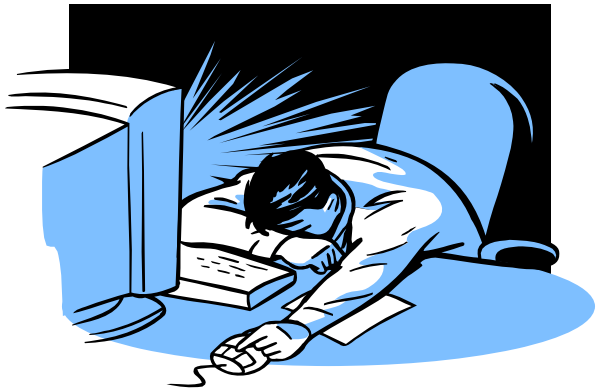
- Process stops working
- Works correctly until time of failure
- “Good Failure” - Detectable
- Monitor process
- Restart crashed process



University of West Florida

Response Failure

- **Process continues operating**
- **Produces incorrect results**
- **“OK failure” – Still detectable**
- **Monitor process**
- **Can’t distinguish between**
  - Slow response
  - Lost request
  - Lost reply
- **Terminate failed process**
- **Start replacement process**

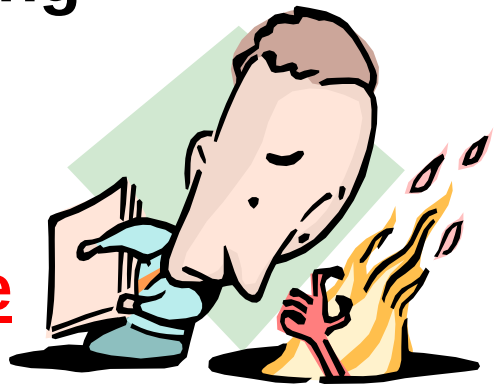


Byzantine Failure

- Process continues executing
- Produces arbitrary results
- … at arbitrary times
- "Bad failure" - Not detectable
- Monitoring doesn't help
- Results could be valid
- Can't distinguish from correctly functioning process
- Exploit code could alter results

  `target=30º27′18.17″,87º12′33.55″`

- Worst-case scenario



University of West Florida

MultiLayer Security Model

Mutation Engine

Executable code generated by mutation engine

- Alters structure of executable code
- Retains functionality of original algorithm

Mitigates Byzantine failures

- Use code mutation
- Doesn't *prevent failures*
- Changes Byzantine → Crash | Response

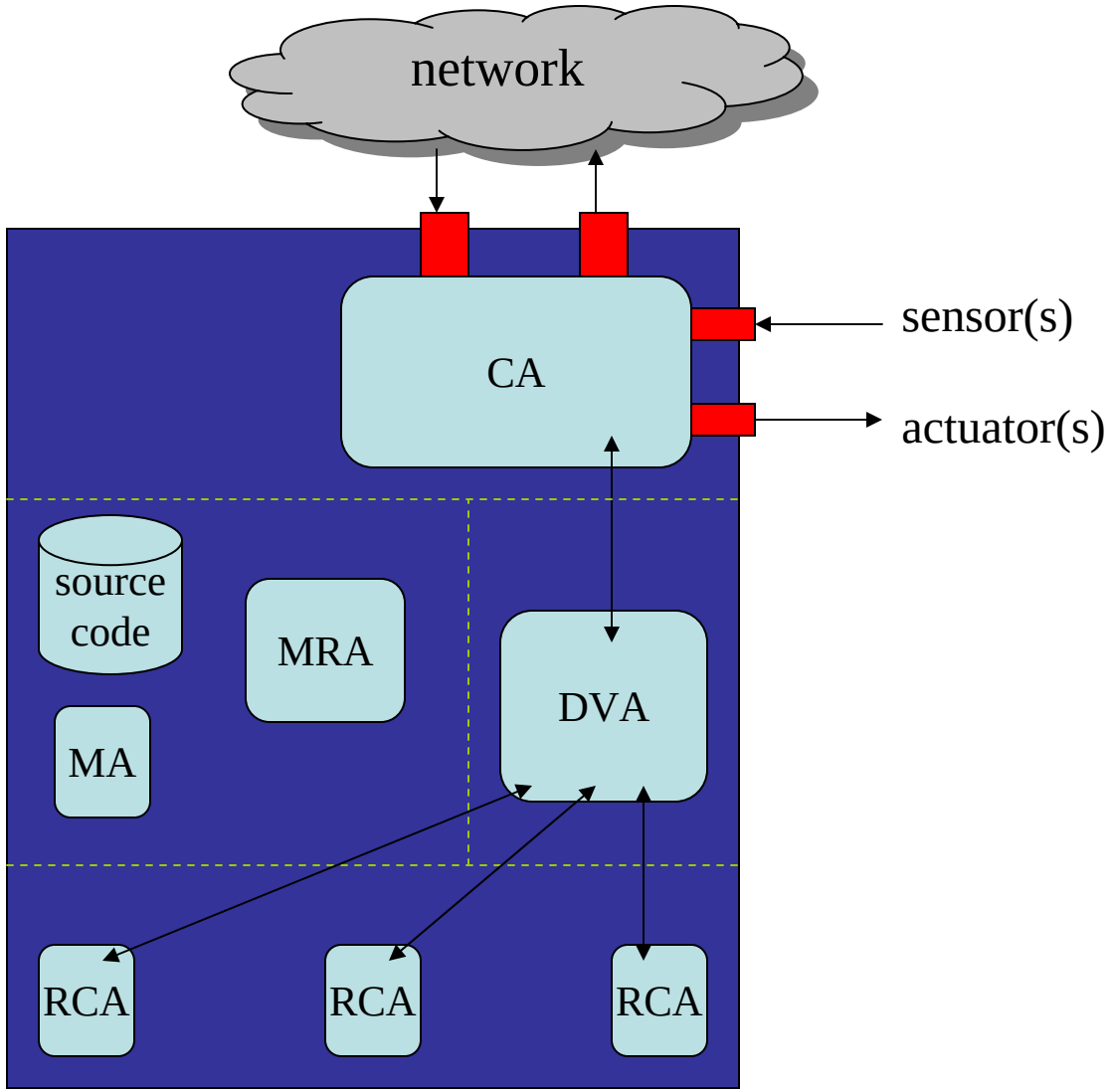
University of West Florida

MultiLayer Security Model

- Filters obvious attacks
- Contains impact of successful attacks
- Quickly recovers from attack
- Detects attacks before consequences are realized
- Prevents recurrence of Byzantine failure

University of West Florida

MLSM – Overview



Agent Layers

MRA – Monitor/Resurrection Agent

- Creates and monitors other agents

CA – Communication Agent

- Communication with external sources
- Authentication and validation

DVA – Distribution/Voting Agent

- Duplicates input to replicated agents
- Calculates majority vote for results

RCA – Replicated Computation Agent

- Performs tasks of original agent
- Only internal communication

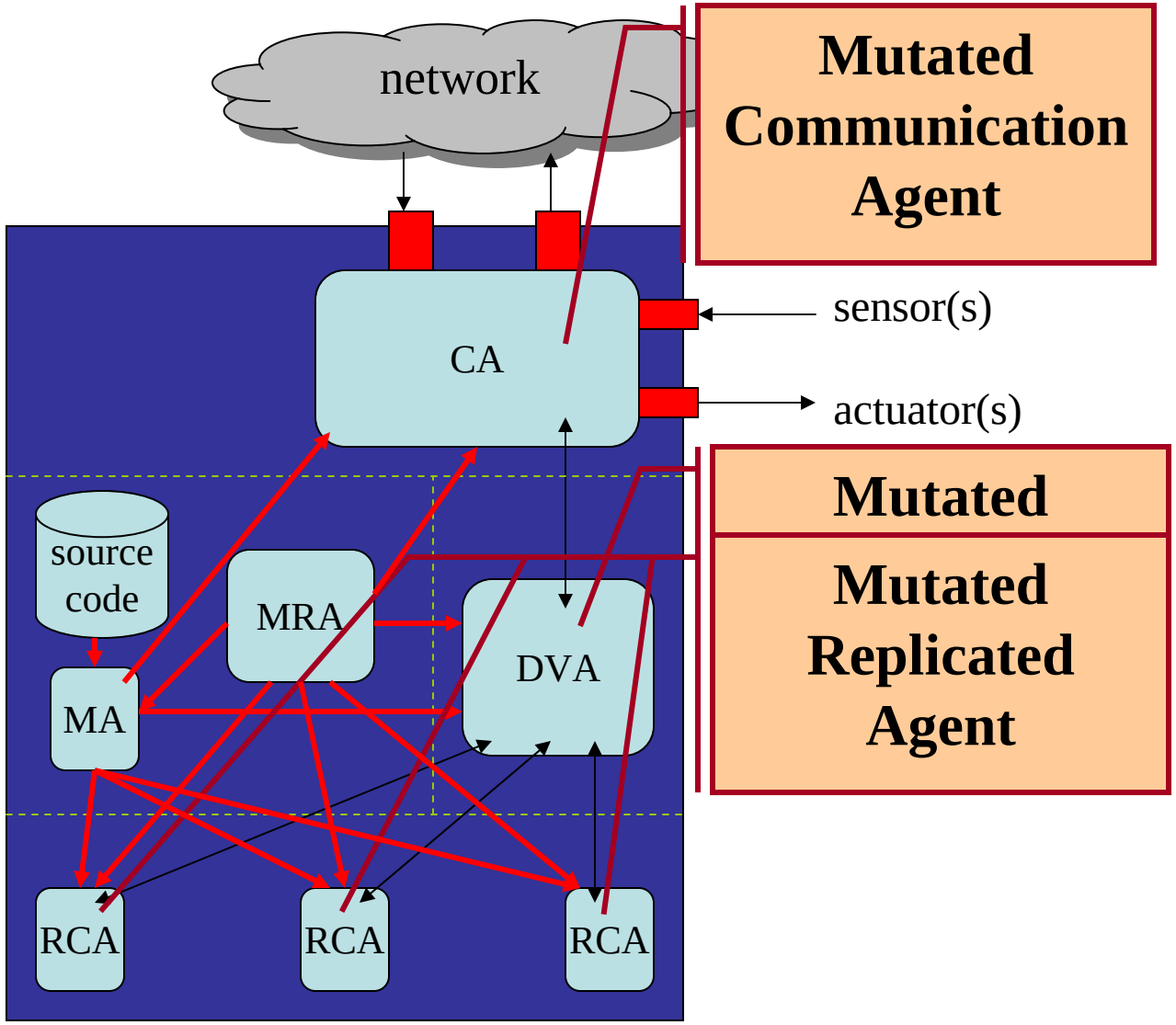
MLSM - Control Security

MRA – Monitor/Resurrection Agent

- Creates and monitors other agents
- Code in FPGA, PROM, etc.
- Uses Mutation Agent to mutate source code
- Creates executing mutants
- Manages communication endpoints (sockets)
- Monitors children for crash failure
- “Ping”s children to monitor for response failure
- Kills and restarts failed children

University of West Florida

MLSM – Initialization



MLSM - Control Security

CA – Communication Agent

- Communication with external sources
- Authentication of source
- Preliminary data validation
- Small, verifiable code
- Internal communication via local (Unix) sockets

University of West Florida

MLSM - Control Security

DVA – Distribution/Voting Agent

- Duplicates input to replicated agents
- Calculates majority vote for results
- Extended data validation
- Only communication via local (Unix) sockets
- No external communication facilities
- Small, verifiable code
- Out-of-bounds response (time or value) from RCA is reported to MRA as failed RCA

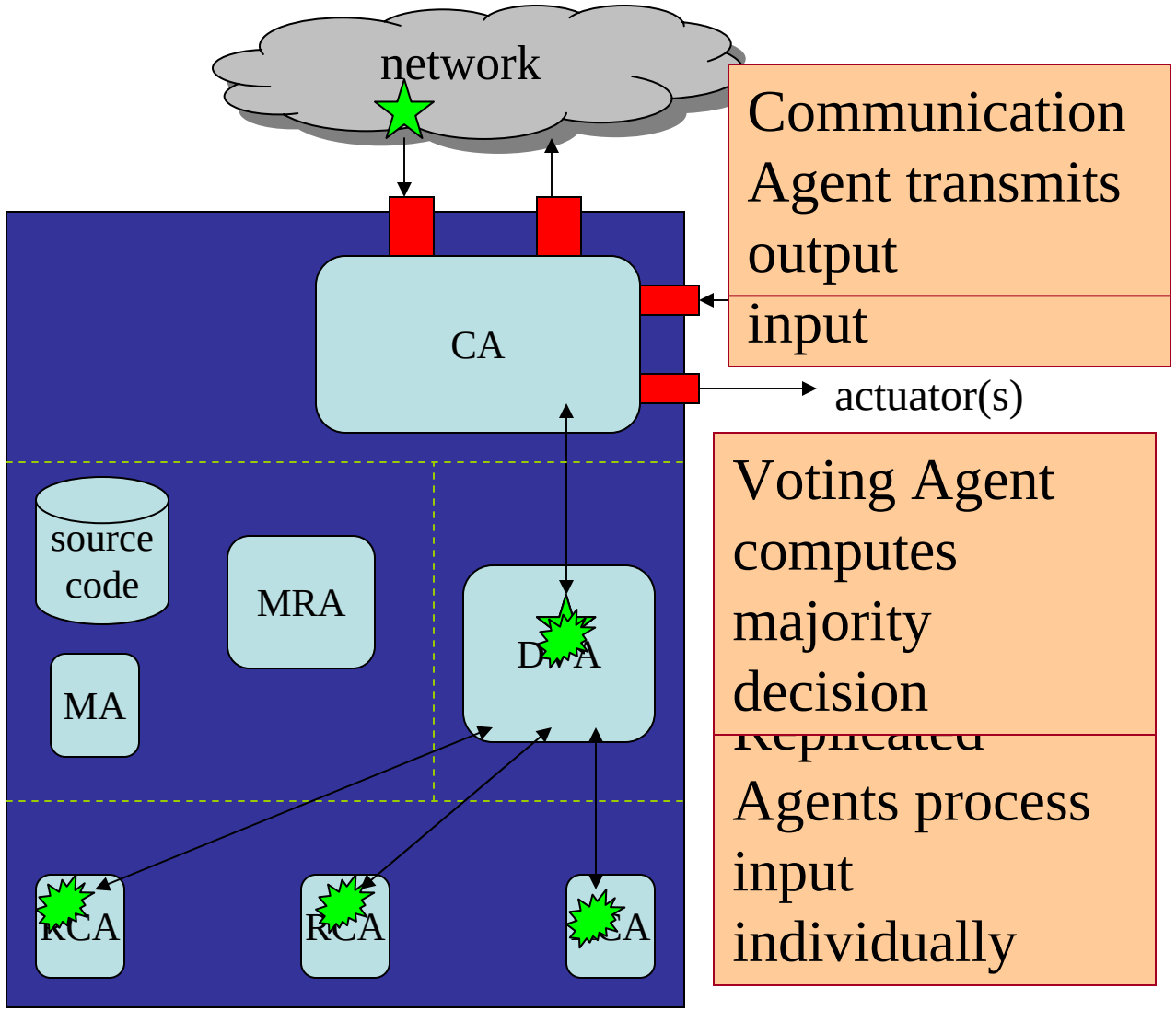
University of West Florida

MLSM - Control Security

RCA – Replicated Computation Agent

- Performs tasks of original agent
- Only internal communication
- Communication via local (Unix) sockets
- Each replica is mutated
- Different, random mutation each time
- Properly functioning ➔ same answer from each

MLSM – Normal Operation



MLSM – Vulnerabilities

External attacks (monitored & resurrectable)

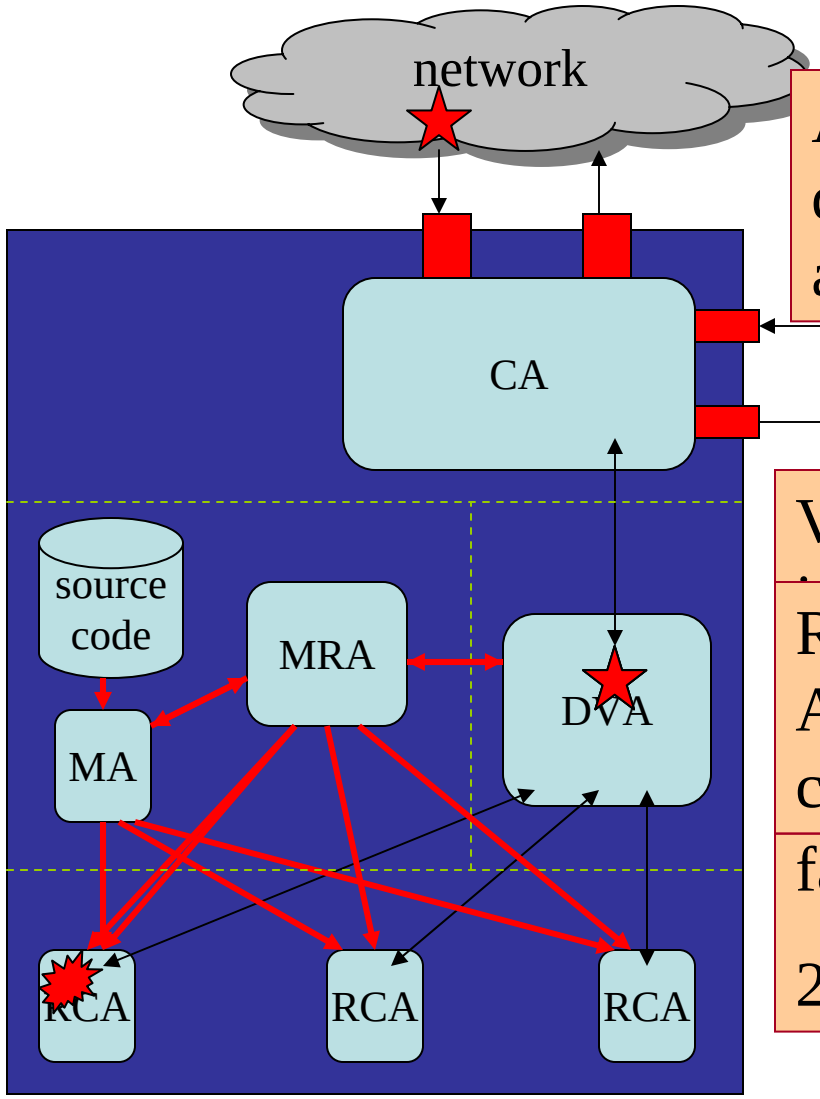
- CA vulnerable (verified, mutated)
- DVA less vulnerable (verified, mutated)
- RCAs protected through mutation and replication

Internal attacks (monitored & resurrectable)

- CA vulnerable – monitored for death
- DVA vulnerable – monitored for death
- RCAs protected through mutation and replication
- MRA less vulnerable – FPGA, EPROM, …

University of West Florida

MLSM – Under Attack



Assume CA doesn't prevent attack

Voting Agent

Resurrection Agent reinstates commo channels

failure

2 in crash failure

MLSM – Mutation Advantages

- Each mutant is structurally different
- Examination of executable doesn't reveal structure of other mutants
- Successful attack (buffer overflow / crafted input) only relevant to single mutant
- Other mutants "resistant" to attack
- Attack can't cause arbitrary code execution in a majority
- Replicated, mutated agents provide redundancy
- Voting prevents attack from propagating to external entities

University of West Florida

MLSM – Comparison

Reach of consequences

- Original: Cascading impacts
- MLSM: Contained impacts

Survivability

- Original: Questionable
- MLSM: Probable

Detectability of Byzantine failure

- Original: Delayed detection
- MLSM: Immediate detection

Recoverability

- Original: Unknown
- MLSM: Rapid recovery

MLSM – Research Status

Prototype system created

Handles buffer overflow attacks

Currently testing efficiency and robustness with RTDS with Florida State University

Plans:

- Extend mutation for other methods of attack
- Apply MLSM to other control systems
- Investigate incorporation with SCADA framework
- Apply MLSM to mobile agents
- Performance improvements to move out of prototype stage

University of West Florida